?????? ??????? ? ??

Определение сервиса на порту

Анализ баннера

У нас есть открытые порты. Но необходимо узнать, что за сервис крутится на нем. Часто сервисы публикуют т н баннер - информацию о себе. Есть сервисы, хранящие данные о сайтах, типа https://shodan.io https://zoomeye.org которые хранят базу данных. Это называется пассивным сканированием.

Естественно администраторы могут изменять баннер.

Netcat может предоставить данную информацию.

```
nc 172.16.10.11 -v 21
172.16.10.11: inverse host lookup failed: Unknown host
(UNKNOWN) [172.16.10.11] 21 (ftp) open
220 (vsFTPd 3.0.5)

nc 1.1.1.1 -v 22
some.address.ru [1.1.1.1] 22 (ssh) open
SSH-2.0-OpenSSH_9.6p1 Ubuntu-3ubuntu13.13
```

Ключи nc:

Флаг	Значение	Пример
`-l`	Слушать (listen) — запускает `nc` в режиме сервера (ожидание входящих подключений)	`nc -l -p 4444`
`-p <порт>`	Указать порт (для прослушивания или исходного подключения)	`nc -l -p 4444`
`-v`	Подробный режим (verbose), показывает процесс соединения	`nc -vz 8.8.8.8 53`
`-vv`	Очень подробный (ещё больше информации)	`nc -zvv 8.8.8.8 53`
`-z`	Сканирование портов (zero-I/O mode: проверка доступности портов без передачи данных)	`nc -zv 192.168.0.1 20-80`
`-u`	Использовать UDP вместо TCP	`nc -u 192.168.0.10 123`

Флаг	Значение	Пример
`-n`	Не использовать DNS (работать только с IP, не пытаться резолвить имена)	`nc -vz -n 192.168.0.1 80`
`-w <сек>`	Таймаут соединения	`nc -vz -w 3 8.8.8.8 53`
`-q <сек>`	Закрыть соединение после EOF через указанное время	`echo hi
`-k`	Продолжать слушать после разрыва соединения (серверный режим)	`nc -lk -p 4444`
`-e <программа>`	Запуск программы после подключения (опасный флаг!, часто отключён в безопасных сборках `nc`)	`nc -l -p 4444 -e /bin/bash`

Сбор информации о баннере:

```
#!/bin/bash

FILE="$1"
PORT="$2"

while read -r ip; do
  res=$(echo -e "\n" | nc -v "${ip}" -w 1 "${PORT}" 2> /dev/null)
  if [[ -n "${res}" ]]; then
    echo "Service: ${ip}:${PORT}"
    echo "Banner: ${res}"
  fi
done < "${FILE}"
```

Анализ http ответа сервера

При помощи curl с помощью метода head можно получить информацию об http сервере.

```
curl --head
172.16.10.10:8081



HTTP/1.1 200 OK
Server: Werkzeug/3.0.1 Python/3.12.3
Date: Sat, 30 Aug 2025 08:41:02 GMT
Content-Type: text/html; charset=utf-8
```

```
Content-Length: 7176
Connection: close
```

Приложение whatweb предоставляет расширенную информацию об http сервере. Синтаксис:

```
whatweb 172.16.10.10:8081
http://172.16.10.10:8081 [200 OK] Country[RESERVED][ZZ], HTML5,
HTTPServer[Werkzeug/3.0.1 Python/3.12.3], IP[172.16.10.10],
Python[3.12.3], Title[Menu], Werkzeug[3.0.1], X-UA-Compatible[ie=edge]

whatweb 172.16.10.10:8081 --log-json=/dev/stdout --quiet | jq
# в формате JSON
```

Т е скомбинировав данные, полученные после сканирования, можно определить некоторые стартовые позиции.

При определении версии сервиса бывает неточным. Дополнительные способы определения версии:

- Подсчет контрольных сумм статичных файлов, для сравнения их с файлами определенной версии (favicon, js код, изображения)
- Исследование изменений в коде и изучение патчей, которые видны из кода веб страниц, сравнение их с версиями кода в репозиториях
- Поиск функций отладки или специальных страниц, раскрывающих информацию о ПО

Также есть базы знаний и инструменты:

- База знаний Common Vulnerabilities and Exposures - https://cve.mitre.org/
- Платформа Vulners - https://vulners.com/
- База данных эксплойтов от Offensive Security - https://www.exploit-db.com/
- Платформа управления уязвимостями и анализа угроз - https://vuldb.com/

Также могут помочь поисковые движки (google.com), публичные репозитории (github.com), блоги разработчиков ПО, китайский сегмент интернета, и пр.

Получение информации об операционной системе

Способ формирования TCP ответа несколько отличаются для разных ОС. Можно определить примерный тип ОС и иногда версию ядра.

Опция -O позволяет проанализировать данную информацию.

```
sudo nmap -O -iL 172-16-10-scanning-hosts.txt

Nmap scan report for 172.16.10.11
Host is up (0.000038s latency).
Not shown: 998 closed tcp ports (reset)
PORT   STATE SERVICE
21/tcp open  ftp
80/tcp open  http
MAC Address: F6:F2:1D:05:71:02 (Unknown)
Device type: general purpose
Running: Linux 4.X|5.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
OS details: Linux 4.15 - 5.19, OpenWrt 21.02 (Linux 5.4)
Network Distance: 1 hop
```

Получение данных:

```
#!/bin/bash

HOSTS="$*"

if [[ "${EUID}" -ne 0 ]]; then
  echo "Run script with sudo."
  exit 1
fi

if [[ "$#" -eq 0 ]]; then
  echo 'Need at least one IP'
  exit 1
fi

nmap_scan=$(sudo nmap -O ${HOSTS} -oG -)
while read -r line; do
  ip=$(echo "${line}" | awk '{print $2}')
  os=$(echo "${line}" | awk -F'OS: ' '{print $2}' | sed 's/Seq.*//g')
  if [[ -n "${ip}" ]] && [[ -n "${os}" ]]; then
    echo "IP: ${ip} OS: ${os}"
  fi
done <<< "${nmap_scan}"
```

Также можно проанализировать версию, обратившись к 445 порту (Windows, Linux).

```
msf > use auxiliary/scanner/smb/smb_version
msf auxiliary(smb_version) > set RHOSTS 192.168.1.200-210
RHOSTS => 192.168.1.200-210
msf auxiliary(smb_version) > set THREADS 11
THREADS => 11
msf auxiliary(smb_version) > run
```

В случае подключенной базы, информация будет в hosts.

Скрытое сканирование

При помощи модуля auxiliary/scanner/ip/ipidseq можно найти машины со старой уязвимостью в нумеровании TCP пакетов, и затем использовать их как зомби.

```
msf > use auxiliary/scanner/ip/ipidseq
msf auxiliary(ipidseq) > show options

Module options (auxiliary/scanner/ip/ipidseq):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   INTERFACE                   no        The name of the interface
   RHOSTS                      yes       The target address range or CIDR identifier
   RPORT      80               yes       The target port
   SNAPLEN    65535            yes       The number of bytes to capture
   THREADS    1                yes       The number of concurrent threads
   TIMEOUT    500              yes       The reply read timeout in milliseconds

msf auxiliary(ipidseq) > set RHOSTS 192.168.1.0/24
RHOSTS => 192.168.1.0/24
msf auxiliary(ipidseq) > set THREADS 50
THREADS => 50
msf auxiliary(ipidseq) > run

[*] 192.168.1.1's IPID sequence class: All zeros
[*] 192.168.1.2's IPID sequence class: Incremental!
[*] 192.168.1.10's IPID sequence class: Incremental!
[*] 192.168.1.144's IPID sequence class: Incremental!
```

Incremental означает возможность использования. Сканируем от имени зомбарей:

```
msf auxiliary(ipidseq) > nmap -Pn -sI 192.168.1.109 192.168.1.114
[*] exec: nmap -Pn -sI 192.168.1.109 192.168.1.114

Starting Nmap 5.00 ( http://nmap.org ) at 2009-08-14 05:51 MDT
Idle scan using zombie 192.168.1.109 (192.168.1.109:80); Class: Incremental
Interesting ports on 192.168.1.114:
Not shown: 996 closed|filtered ports
PORT STATE SERVICE
135/tcp open msrpc
139/tcp open netbios-ssn
445/tcp open microsoft-ds
3389/tcp open ms-term-serv
MAC Address: 00:0C:29:41:F2:E8 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 5.56 seconds
```

Определение MSSQL

В metasploit много сканеров на различные сервисы, список

```
use auxiliary/scanner/
```

Revision #4
Created 2 October 2025 07:08:11 by Admin
Updated 2 October 2025 08:29:41 by Admin